

УДК 621.311

М.Ю.ЧУКРЕЕВ, канд. техн. наук

Учреждение Российской академии наук Институт социально-экономических и энергетических проблем Севера Коми научного центра УрО РАН, г.Сыктывкар

ВОЗМОЖНОСТИ ПОВЫШЕНИЯ ВЫЧИСЛИТЕЛЬНОЙ ЭФФЕКТИВНОСТИ ПРОГРАММНО-ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА «ОРИОН-М-ЗСПМ»

Исследуется эффективность ускорения процесса вычислений применительно к программно-вычислительному комплексу «ОРИОН-М-ЗСПМ».

Досліджується ефективність прискорення процесу обчислень стосовно до програмно-обчислювального комплексу «ОРИОН-М-ЗСПМ».

Determined by the effectiveness of different approaches accelerate the process of calculation software and computing system "ORION-M-ZSPM".

Ключевые слова: компилятор, надежность, процедуры OpenCL, параллельные вычисления.

В моделях оценки показателей балансовой надежности используются различные математические методы, переведенные в машинные алгоритмы. Независимо от используемого языка программирования всегда есть возможность изменить код с целью ускорения процесса вычислений. Это позволяет снять ряд ограничений и допущений в используемых математических моделях, улучшая достоверность получаемых результатов. Способствует этому и принятие 16.05.2011 г. комплексного пакета документов о технологических правилах работы электроэнергетических систем (ТПР ЭЭС) общеобязательного (нормативного) характера, разработанных по поручению Министра энергетики РФ ОАО «СО ЕЭС».

Вопросам оценки балансовой надежности (БН) в проекте ТПР ЭЭС уделено достаточное внимание. Данная оценка на предстоящий планируемый период должна осуществляться ОАО «СО ЕЭС» (в изолированно работающих территориальных энергосистемах – субъектом оперативно-диспетчерского управления) по каждой концентрированной энергосистеме¹, территориальной энергосистеме, ОЭС и ЕЭС в целом. Обоснованным является пункт ТПР, касающийся необходимости применения математических моделей для оценки БН ЭЭС. Все это в совокупности требует проведения научно-исследовательских работ по обобщению существующего опыта разработки таких моделей, учету влияющих на БН факторов, сравнительной вычислительной эффективности и т.д. По-

¹ Концентрированным называют энергосистему, внутри которой отсутствуют ограничения на передачу мощности по линиям электропередачи из одной зоны в другую.

является необходимость проведения расчетов по оптимизации распределения резервов мощности как с позиции надежности, так и экономической эффективности.

Все эти факторы приводят к существенному увеличению объемов вычислений, а следовательно, и затрачиваемого времени. При некоторых условиях оно может вырасти в десятки раз. Поэтому актуальность разработок по ускорению процесса расчета будет только расти. Рассмотрим это для программно-вычислительного комплекса оценки показателей БН «ОРИОН-М-ЗСПМ», разработанного в отделе энергетики ИСЭиЭПС Коми НЦ УрО РАН.

Известно, что скорость исполнения любого приложения можно увеличить несколькими способами. Наиболее распространенные – применение соответствующих компиляторов, оптимизированных на работу в определенных системах, и использование параллельных вычислений. Первый вариант без труда можно использовать в компьютерных системах оснащенных центральными процессорами intel. В этом случае выбирается компилятор нужного языка производства той же компании. С центральными процессорами AMD такой вариант менее эффективен (насколько известно автору). Во втором случае оптимально использовать OpenCL – новый стандарт для разработки приложений гетерогенных систем [1]. Этот стандарт используется для написания приложений, которые должны исполняться в системе, где установлены различные по архитектуре процессоры, графические ускорители и платы расширения. Таким образом, отпадает необходимость использовать разные алгоритмы для систем, основанных на платформах intel и AMD.

В нескольких словах опишем основные принципы дизайна OpenCL систем. Не вдаваясь в излишние на данном уровне изложения подробности, опишем четыре модели, на которых держится стандарт: модель платформы, модель исполнения, модель памяти и модель программирования.

OpenCL – индустриальный стандарт, рожденный в 2008 г. в результате взаимодействия специалистов компании Khronos Group с разработчиками программного обеспечения, вендорами аппаратных решений (в том числе мобильных платформ) и производителями процессоров различных типов и назначений. Центральная идея OpenCL – предоставить программисту универсальный инструмент для использования всех вычислительных мощностей современных вычислительных систем. Основную идею, положенную в основу этого стандарта [2], можно описать так: написав однажды программу с использованием OpenCL, можно запускать ее практически на любой вычислительной системе: телефонах, графических картах, ускорителях и т.п. Данный стандарт явился важным

шагом в развитии параллельного программирования для гетерогенных систем¹.

Созданный на OpenCL программно-вычислительный комплекс может использовать имеющиеся ресурсы вычислительной системы следующим образом: определить доступные ресурсы в гетерогенной системы и выбрать подходящие; создать последовательность инструкций, которые будут выполняться на ресурсах; собрать результаты вычислений.

Для создания OpenCL приложения необходимо его продумать с четырех точек зрения.

Модель платформы (platform model) дает высокоуровневое описание гетерогенной системы. Центральным элементом данной модели выступает понятие хоста (основной процедуры, host) – первичного устройства, которое управляет OpenCL-вычислениями и осуществляет все взаимодействия с пользователем (рис.1). Это ядро программы, оно представлено в единственном экземпляре, в то время как OpenCL-устройства (devices), на которых выполняются OpenCL-инструкции, могут быть представлены во множественном числе. OpenCL-устройством может быть CPU, GPU, DSP или любой другой процессор в системе, поддерживающийся установленными в системе OpenCL-драйверами.

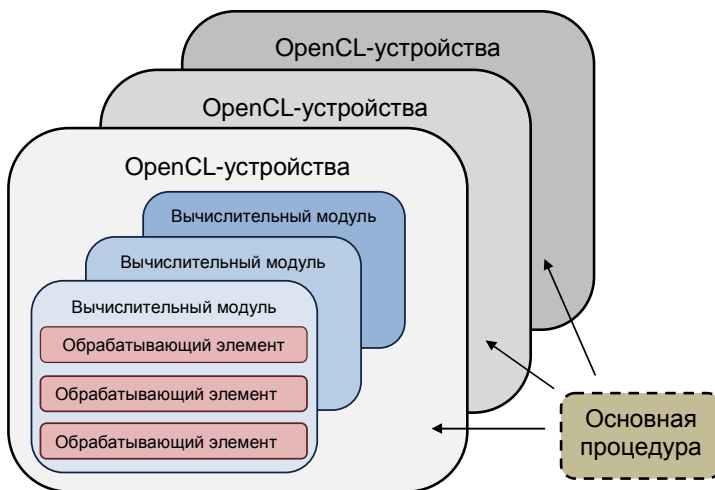


Рис.1 – Схематическое представление OpenCL-платформы

¹ Гетерогенная система (от греч. ἕτερος — разный; γένω — рождать) – неоднородная система, состоящая из однородных частей.

OpenCL-устройства логически делятся моделью на вычислительные модули (compute units), которые в свою очередь делятся на обрабатывающие элементы (processing elements). Вычисления на OpenCL-устройствах в действительности происходят на обрабатывающих элементах.

Модель вычислений (execution model) описывает абстрактное представление того, как потоки инструкций выполняются в гетерогенной системе.

С основной процедурой неразрывно связано понятие хостовой (основной) программы (host program) – программного кода, выполняющегося только на хосте. OpenCL не указывает, как именно должна работать хостовая программа, а лишь определяет интерфейс взаимодействия с OpenCL-объектами. С точки зрения модели вычислений OpenCL-приложение состоит из хостовой программы (выполняется основной процедурой) и набора ядер (kernels). OpenCL-ядро в общем виде представляет собой функцию, написанную на языке OpenCL C (подмножество языка ISO C'99) и скомпилированную OpenCL-компилятором.

Ядро создается в основной программе и затем с помощью специальной команды ставится в очередь на выполнение в одном из OpenCL-устройств. Во время выполнения упомянутой команды (OpenCL Runtime System) создается целочисленное пространство индексов (integer index space), каждый элемент которого обозначен глобальным идентификатором (global ID). Каждый экземпляр ядра выполняется отдельно для каждого значения глобального идентификатора. Экземпляр ядра носит название рабочего элемента (work-item). Таким образом, каждый рабочий элемент однозначно определяется своим глобальным идентификатором.

Множество всех рабочих элементов разбивается на группы, которые называются рабочими группами (work-group). С каждой рабочей группой сопоставляется свой уникальный идентификатор (work-group ID). Все рабочие элементы в одной рабочей группе идентифицируются уникальным в пределах своей группы номером: local ID. Таким образом, каждый элемент определяется как по уникальному global ID, так и по комбинации идентификационного номера рабочей группы и номера в пределах этой группы.

Все рабочие элементы в пределах одной группы выполняются параллельно на обрабатывающих элементах одного вычислительного модуля OpenCL-устройства. Взаимодействие между хостом и OpenCL-устройством происходит посредством команд, помещенных в командную очередь (command-queue). Данные команды ожидают в командной очереди своего выполнения на OpenCL-устройстве. Командная очередь создается хостом и сопоставляется одному OpenCL-устройству после

того, как будет определен контекст. Команды делятся на выполняющие ядра, управляющие памятью и синхронизирующие выполнения команд в очереди. Команды могут выполняться последовательно (in-order execution) или непоследовательно (out-of-order execution). Второй вариант организации очередей поддерживается не всеми платформами.

Модель памяти (memory model) описывает набор регионов памяти и манипулирование ими во время проведения вычислений.

OpenCL-объекты, инкапсулирующие регионы памяти, носят название объектов памяти (memory objects). Объекты памяти бывают двух типов: буферные объекты (buffer objects) и объекты изображения (image objects).

Стандарт описывает пять различных регионов памяти (рис.2) из которых нас интересуют только три:

- память хоста (host memory), доступная лишь с хоста;
- глобальная память (global memory), определенная в памяти, доступной на чтение и запись для всех рабочих элементов во всех группах. В подавляющем большинстве случаев глобальная память самая медленная, поэтому использовать ее стоит по минимуму;
- локальная память (local memory), доступная лишь в пределах одной рабочей группы. Все элементы в данной рабочей группе могут как читать, так и писать в данный регион памяти одновременно.

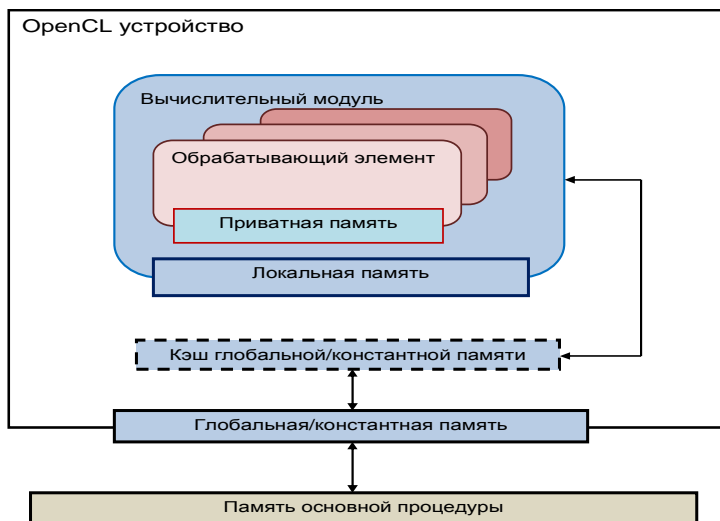


Рис. 2 – Схематическое представление нескольких уровней памяти в OpenCL

Модель программирования (programming model) описывает варианты переноса алгоритма на гетерогенные вычислительные ресурсы. OpenCL определяет два типа модели программирования: параллелизм по данным (data parallelism) и параллелизм по заданиям (task parallelism). Параллелизм по данным организован вокруг структур данных: каждый элемент определенной хостом структуры данных обновляется одновременно (параллельно) копиями одного и того же OpenCL-ядра. Модель параллелизма по данным наиболее естественная модель программирования для OpenCL. Задача программиста здесь описать решаемую задачу в терминах упомянутой структуры данных.

В нашем случае между несколькими рабочими элементами необходимо взаимодействие, поэтому алгоритм проектируется с учетом разбиения множества всех элементов на некоторое количество рабочих групп, в рамках которых несколько элементов могут осуществлять такое взаимодействие. Взаимодействие осуществляется через чтение/запись локальных регионов памяти.

Ввиду ориентированности технологии OpenCL на широкий круг устройств необходимо помнить, что OpenCL не поддерживает механизмы синхронизации между несколькими элементами из разных рабочих групп. Кроме того, все рабочие элементы в пределах группы выполняются одновременно, но при этом несколько групп могут выполняться параллельно, если устройство такую возможность предоставляет. Таким образом, получается иерархическая модель параллелизма по данным – внутри каждой группы и между различными группами.

Параллелизм по заданиям предполагает процедуру, выполняемую как единственную на рабочем элементе. При этом вместе с таким заданием (процедурой) в устройстве могут одновременно выполняться и другие элементарные задания.

Также данный тип модели программирования может возникнуть, когда несколько команд запуска ядер помещаются в очередь, в которой запуск происходит сразу же, как только команда в нее поместилась. В ряде случаев это позволяет увеличить степень использования OpenCL-устройств, предоставляя системе возможность самостоятельно планировать запуск множества различных заданий.

Третий вариант параллелизма по заданиям возникает, когда задания зависимы между собой и объединяются в граф с применением OpenCL-событий. Одни команды, помещенные в очередь, могут генерировать события, а другие – ожидать этих событий, чтобы начать свое выполнение.

Первые два варианта параллелизма неплохо вписываются в решаемые задачи. Какой из них окажется оптимальным покажет время, но в

любом случае, применение алгоритмов, используемых в OpenCL приложениях, заслуживает повышенного внимания в силу своей универсальности. Скорость расчета такого продукта хорошо масштабируется, а ограничения по привязке к определенной платформе отсутствуют.

Для определения эффективности использования каждого метода использовалась 51 узловая схема ЕЭС России [3], представленная на рис.3. Следует помнить, что конкретные цифры сильно зависят от используемого программного и аппаратного обеспечения, но о приближенных соотношениях представление дает. Расчеты проводились путем 10-кратного выполнения программно-вычислительного комплекса при неизменных параметрах. Итоговым берется среднее значение по 10 экспериментам. Разница между результатами однотипных расчетов обусловлена различным уровнем загрузки аппаратной и программной

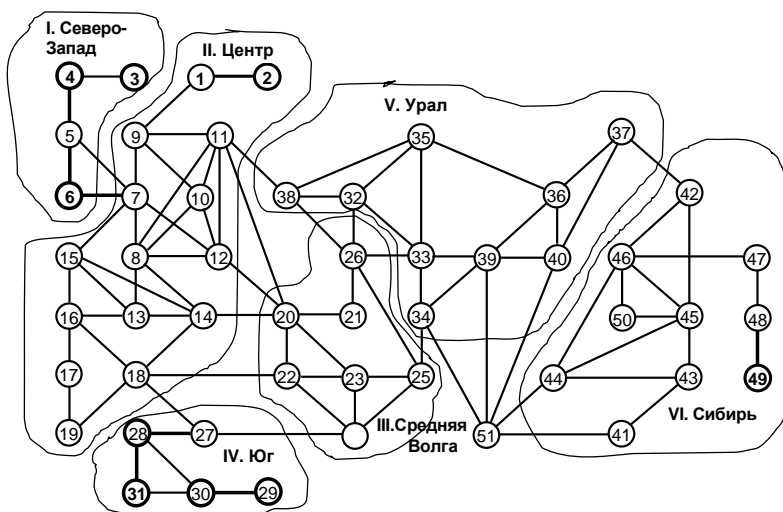


Рис.3 – Расчетная схема ЕЭС России при условном разбиении ее на ЗСПМ:

1 – Архангельск; 2 – Коми; 3 – Кола; 4 – Карелия; 5 – С-Петербург, Ленинградская обл.; 6 – Псков, Новгород; 7 – Тверь; 8 – Москва, Московская обл.; 9 – Вологда; 10 – Ярославль; 11 – Кострома; 12 – Владимир, Иваново; 13 – Тула, Калуга; 14 – Рязань; 15 – Смоленск; 16 – Орел, Брянск; 17 – Курск, Белгород; 18 – Липецк, Тамбов; 19 – Воронеж; 20 – Н.Новгород; 21 – Чувашия, Мари-Эл; 22 – Мордовия, Пенза; 23 – Ульяновск; 24 – Саратов; 25 – Самара; 26 – Татарстан; 27 – Волгоград, Астрахань, Калмыкия; 28 – Ростов; 29 – Дагестан, Чечня, Кабардино-Балкария; 30 – Ставрополь, Ингушетия, Карачаево-Черкесская; 31 – Кубань; 32 – Удмуртия; 33 – Башкирия; 34 – Оренбург; 35 – Пермь; 36 – Екатеринбург, Свердловская обл.; 37 – Тюмень; 38 – Киров; 39 – Челябинск; 40 – Курган; 41 – Омск; 42 – Томск; 43 – Новосибирск; 44 – Алтай; 45 – Кузбасс; 46 – Красноярск; 47 – Иркутск; 48 – Бурятия; 49 – Чита; 50 – Хакассия, Тыва; 51 – Северный Казахстан.

составляющей вычислительной машины в разные моменты времени. Полученные результаты сведены в таблицу. Расчет проводился на центральном процессоре и графической карте, аналогичной для обеих платформ. Работа над полноценным использованием этого алгоритма только начата, но уже сейчас понятно, что эффект от его применения существенен, и развитие программно-вычислительного комплекса «ОРИОН-М-ЗСПМ» по этому направлению кажется нам наиболее обоснованным.

Коэффициенты ускорения расчетов при разных вариантах программы

Вариант модификации кода	Intel	AMD
1. Компилятор оптимизированный под соответствующий центральный процессор без многопоточности	1,93	1,24
2. Компилятор оптимизированный под соответствующий центральный процессор с учетом многопоточности	2,77	1,35
3. OpenCL	4,19	4,06

1. www.nvidia.com/object/cuda_opengl.html.

2. <http://www.khronos.org/registry/cl/sdk/1.1/docs/man/xhtml.html>.

3. Чукреев Ю.Я., Чукреев М.Ю. Обеспечение надежности электроэнергетических систем при управлении их развитием в условиях реформирования электроэнергетики. – Сыктывкар, 2009. – 44 с.

Получено 07.11.2011

УДК 697.34

Д.О.БОЦУЛА

Державна інспекція з енергетичного нагляду за режимами споживання електричної і теплової енергії, м.Київ

А.І.ЗАМУЛКО, канд. техн. наук

Національний технічний університет України "КПІ", м.Київ

С.О.ШАПКА

ТОВ "Біолінк", м.Київ

ОЦІНКА ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ ЕНЕРГЕТИЧНОГО НАГЛЯДУ В ІНСПЕКЦІЇ ДЕРЖЕНЕРГОНАГЛЯДУ

Аналізуються існуючі показники оцінки ефективності впровадження інноваційних проєктів в державних установах, основними завданнями яких є здійснення нагляду в сферах господарської діяльності.

Анализируются существующие показатели оценки эффективности внедрения инновационных проектов в государственных учреждениях, основными задачами которых есть осуществление надзора в сфере хозяйственной деятельности.

The analysis of existing indicators to assess the effectiveness of innovative projects in public institutions, whose main tasks are to supervise in the field of economic activity, is.